## *Amendments to the Specification*

Please amend the specification as indicated.

Please amend the paragraph starting on page 7, line 22, as follows:

FIG. 2 illustrates an exemplary computer system 212 comprised of a system bus 200 for communicating information, one or more central processors 201 coupled with the bus 200 for processing information and instructions, a computer readable volatile memory unit 202 (e.g., random access memory, static RAM, dynamic RAM, etc.) coupled with the bus 200 for storing information and instructions for the central processor(s) 201, a computer readable non-volatile memory unit 203 (e.g., read only memory, programmable ROM, flash memory, EPROM, EEPROM, etc.) coupled with the bus 200 for storing static information and instructions for the processor(s).

Please amend the paragraph starting on page 9, line 11, as follows:

The vector register file is comprised of 32 64-bit general purpose registers 306 through 310. The general purpose registers 306 through 310 are visible to the programmer and can be used to store intermediate results. The preferred embodiment of the present invention uses the floating point registers ~~(FGR)~~ (FPR) of a floating point unit (FPU) as its vector registers.

Please amend the paragraph starting on page 13, line 26, as follows:

Integer vector operations that write to the FPRs clamp the values being written to the target's representable range. That is, the elements are saturated for overflows and ~~under-flows~~ underflows. For overflows, the values are clamped to the largest representable value. For underflows, the values are clamped to the smallest representable value.

Please amend the paragraph starting on page 17, line 19, as follows:

Load Vector Add (ADDL.fmt). According to the ADDL.fmt instruction, the corresponding elements in vectors vt and vs are added and then stored into corresponding elements in the accumulator. Any overflows or underflows in the elements wrap around the accumulator's representable range and then are written into the accumulator ~~706~~ 806.

Please amend the paragraph starting on page 22, line 1, as follows:

A RACL/RACM/RACH instruction followed by WACL/WACH are used to save and restore the accumulator. This ~~save/ restore~~ save/restore function is format independent, either format can be used to save or restore accumulator values generated by either QH or OB operations. Data conversion need not occur. The mapping between element bits of the OB format accumulator and bits of the same accumulator interpreted in QH format is implementation specific, but consistent for each implementation.